



Performance evaluation of In-Deep Class Storage for Flow-Rack AS/RS

Olivier Cardin, Pierre Castagna, Zaki Sari, Nihad Meghelli

► To cite this version:

Olivier Cardin, Pierre Castagna, Zaki Sari, Nihad Meghelli. Performance evaluation of In-Deep Class Storage for Flow-Rack AS/RS. International Journal of Production Research, 2012, 50 (22-24), pp.6775-6791. 10.1080/00207543.2011.624561 . hal-00784338

HAL Id: hal-00784338

<https://hal.science/hal-00784338>

Submitted on 4 Feb 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Performance evaluation of In-Deep Class Storage for Flow-Rack AS/RS

Olivier Cardin ^a, Pierre Castagna ^a, Zaki Sari ^b, Nihad Meghelli ^b

^a *LUNAM Université, IUT de Nantes – Université de Nantes, IRCCyN UMR CNRS 6597 (Institut de Recherche en Communications et Cybernétique de Nantes), 2 avenue du Pr Jean Rouxel – 44475 Carquefou*

^b *Manufacturing Engineering Laboratory of Tlemcen (MELT), B.P 230, 13000 Tlemcen, Algeria*

Corresponding Author : Olivier Cardin : olivier.cardin@irccyn.ec-nantes.fr;
Tel : +33 (0) 228092020 ; Fax : +33 (0) 228092021

Provide short biographical notes on all contributors here if the journal requires them.

Performance evaluation of In-Deep Class Storage for Flow-Rack AS/RS

This article presents a new storage-retrieval method, called In-Deep Class Storage, designed for Flow-Rack AS/RS. If class-based storage is a well known method a lot studied in literature, this method is based on the statement that it is more efficient to dedicate the front layers of each bin to the class of the most popular items, rather than dedicating whole bins close to the drop-off station, as already studied in the literature. Obviously, this idea is not trivial to implement, due to the dynamic behavior of such racks. Thus, two separate algorithms were defined, one for storage, one for retrieval, enabling a dynamic use of our approach, with the only hypothesis of a Pareto distribution of the items demand. This article finally presents a simulation study, designed to compare the performance of random storage and retrieval to the use of the algorithms. This study shows significant improvement of the expected retrieval delay, which was chosen as the main performance indicator.

Keywords: flow-rack automated storage and retrieval systems (flow-rack AS/RS); storage and retrieval heuristic; class-based storage; simulation with discrete events; performance assessment; in-deep class storage

1 Introduction

Automated storage and retrieval systems (AS/RS) have been widely used not only as alternatives to traditional warehouses, but also as a part of advanced manufacturing systems (Lee, 1997). Today, because of the drastic development of B2C (business to consumer), the AS/RS has become an important link in the supply chain. They make it possible to improve the productivity in organizations of manufacture and distribution, in order to reduce the storage cost, to improve the progress of items and storage management. They have a great flexibility of interfacing with other components of FMS (Materiel handling, control) and can manage items of any kind (tools, raw materials, pallets, products, support) with quick responses to maintain the demands of manufacturing, warehousing, and distribution applications. Moreover, they have a minimal use of space (Allen, 1992).

Between 1994 and 2004, there was a significant increase in the number of AS/RSs used in the industrial environments in the United States (Roodbergen and

Vis, 2009). During the last decades several types of AS/RS were developed to treat the growing number of different references of products to store. There exist various types of AS/RS. They are differentiated by their weight, their size and the storable volume of loads. Examples of these are: unit-load, mini-load, deep rack, slipping rack, deep-lane and flow-rack system (Bozer and White, 1984, and Sari *et al.*, 2005).

The flow rack AS/RS is a deep rack made up of a whole of bins whose supports are gravitational conveyors. Two S/R machines are at the two ends, one on a face ensuring storage and the other on the opposite face ensuring retrieval. Both machines are connected by a restoring conveyor inclined in opposite direction making it possible to items to travel from retrieval face to storage face. Consequently storage machine can reach items to restore them. Other storage rack designs are examined nowadays, such as turnover-based storage racks (Yu and De Koster, 2009). This kind of rack enables the use of only one machine for storage and retrieval. These racks are not considered in this study.

This article defines a new storage-retrieval method, called In-Deep Class Storage, designed for Flow-Rack AS/RS. If class-based storage is a very well known method a lot studied in literature, this method is based on the statement that it is more efficient to dedicate the front layers of each bin to the class of the most popular items, rather than dedicating whole bins close to the drop-off station, as already studied in the literature. Obviously, this idea is not trivial to implement, due to the dynamic behavior of such racks. Thus, two separate algorithms were defined, one for storage, one for retrieval, enabling a dynamic use of the method, with the only hypothesis of a Pareto distribution of the demand of items.

After a literature survey of the various approaches of storage in a standard AS/RS, in particular the works on class-based storage, the principle of In-Deep Class

Storage is defined, and a validation study is presented. Then, both the algorithms are presented in detail. This article finally presents a simulation study, designed to compare the performance of random storage and retrieval to the use of the algorithms.

2 Class-based storage in literature

2.1 Performance evaluation of dynamic storage/retrieval algorithms

Numerous algorithms were developed in literature, concerning many different types of AS/RS. For example, Park *et al.* (2003) and Park *et al.* (2006) studied and analyzed the performances of a mini-load AS/RS with 2-classes. Caron *et al.* (1998) and Caron *et al.* (2000) proposed a model of cycle time for a 2-block warehouses (i.e., one middle cross aisle) with a cube-order-index (COI)-based storage.

Ashayeri *et al.* (2002) used a geometric approach to calculate the cycle time for a random storage. Le-Duc and De Koster (Le-Duc and De Koster, 2004), (Le-Duc and De Koster, 2005), (Le-Duc and De Koster, 2007), as well as Roodbergen (2001) estimated the cycle time of a retrieving by batches, i.e. gathering several requests for retrieval in a delivery, in order to reduce the expected retrieval delay. As this paper states that the delivery request of items are treated with FCFS rule, i.e. without trying to batch these requests, these works are not detailed anymore.

Mansouri (1997) and Eldemir *et al.* (2004) developed analytic expressions of cycle time in the case of dedicated storage. In the case of the flow rack AS/RS, the expected retrieval delay for a random storage, i.e. without studied rule of storage of the references, is rather not very powerful (Sari *et al.*, 2005).

Gaouar *et al.* (Gaouar *et al.*, 2005), (Gaouar *et al.*, 2006) developed storage/retrieval heuristic to reduce this expected retrieval delay. The heuristic is developed so as to store the products of same references in the same bin with more close to the drop-off station. This heuristic has allowed a great improvement of the

expected retrieval delay. Thus, considering the example of a system with 1000 storage segments (10x10x10) using random storage, and this same system using the storage/retrieval heuristic for 200 different references, the saving of retrieval delay is on average 40% .

Yu and De Koster (2011) developed a heuristic to optimize the performance in sequencing the retrievals in a multi-deep rack. This heuristic allowed a great improvement compared to the simple First Come First Served rule. In a first approach, this paper only focuses on the problem of single command, a next step being the evaluation of this heuristic on racks using In-Deep Class storage.

2.2 Performance evaluation of class based storage

Eldemi *et al.* (2004) proposed a new analytical model for the AS/RS to determine the cycle time and space required for storage using different storage methods: dedicated, random and class-based. From this model, they deduced that the use of class-based storage requires less storage space than the use of random storage or dedicated storage, and moreover, the storage and retrieval delay are lower.

Muppani and Adil (2008) developed a nonlinear integer programming model combined with a Branch & Bound algorithm to compare class-based storage to dedicated storage, considering the storage space and its cost, handling cost and order picking for class-based storage. The approach presented has demonstrated that there are significant savings in using class-based storage policy, although the dedicated and random policies are commonly used in warehouse storage planning.

Although these results were determined on unit-load racks, they tend to prove the efficiency of class-based storage in any configuration, including deep racks. Next section presents the works related to the geographical location of classes in the racks.

2.3 *Geographical layout of classes*

Hausman *et al.* (1976) were among the first to consider the problem of class-based storage in an AS/RS with simple cycle, by using the ABC curve.

According to the policy of class-based storage, the products in the store are divided into groups based on their frequency of input/output. These groups of products are assigned to a region (class) where the number of regions is equal to the number of groups. The allocation of group of items to the regions is made according to the frequency of input/output. Thus, the region that gives rise to the lowest expected retrieval delay (including restoring) is assigned to the group of articles with the highest frequency. This process continues until the area giving the highest expected retrieval delay is assigned to the group of items with the lowest frequency.

The main contribution of (Hausman *et al.*, 1976), studied deeper in Rosenblatt and Eynan (1989) and Eyan and Rosenblatt (1994), was to determine optimal shape of the bins assignation to classes in order to lower the average retrieval delay by assigning regions of bins to specific classes (Figure 1) in an unit-load rack. This choice explains the position of classes (Figure 1), drawing arcs around the delivery station. In a deep rack, optimality is not valid anymore, as the restoring station is not taken into account. Yu *et al.* (2009b) lead the same type of study on a specific deep rack. Obviously, these results cannot be directly applied in this work, as the class limits cannot be drawn on the faces, but are located in the depth of the rack.

<Please insert Figure 1 here>

Furthermore, Graves *et al.* (1977) showed by simulation that the L-shaped of different regions, for class-based storage, are not necessarily optimal for square racks. Guenov and Raeside (1992) compared three configurations of the shape of zones in an AS/RS. They conclude that the zone configuration for class-based storage depends on

the position of the pickup/drop-off stations from the rack for a unit-load AS/RS, which is also applicable in the case of deep racks.

2.4 Optimal number of classes

Speaking about class storage, several authors dealt with the problem of optimal number of classes to be used in a context of AS/RS storage.

Rosenblatt and Eynan (1989) and Eynan and Rosenblatt (1994) sought to determine the optimal number of classes for an AS/RS using the method of Hausman et al. (1976) presented before. They concluded that it is preferable to use unit-load racks with a Shape factor $b \geq 0.6$ and less than 7 classes. Ashayeri *et al.* (2002) have developed a heuristic approach to determine the number of classes and the location of the storage areas for class-based storage in an AS/RS. The algorithm determines the size and number of zones, and their arrangement on the rack of AS/RS.

Ha and Hwang (1994) have shown that 2-class-based storage assignment policy reduces significantly the storage time and the retrieval delay of both single and double commands over randomized storage assignment policy in a carousel system.

As a conclusion, all these studies show the importance of studying the performance of the methods of storage in the storage rack, especially through the performance criterion of the expected retrieval delay.

3 In-deep class storage

3.1 Studied Flow-Rack

In this study, the pickup station is located at the corner down the front of storage and drop-off station at the corner down the other side. The optimal dwell points, as determined by Peters *et al.* (1996), are the pickup station for the storage machine and mid-point of the rack for retrieval machine.

As shown in Figure 2, a rack has a height H , a length L and depth D . It is composed of bins containing M storage segments arranged in a multi-layer matrix composed of N_l horizontal bins, N_h vertical bins and M layers numbered from 0 (retrieval face) to $M-1$ (storage face). In each storage segment, a single item can be stored.

<Please insert Figure 2 here>

The storage machine and retrieval machine may move simultaneously on two axes (Chebyshev Travel – (Bozer *et al.*, 1987)). These two axes form a plane parallel to both sides of the rack. Storage/ retrieval machines have the same horizontal velocity V_h and the same vertical velocity V_v .

In addition to Figure 2, the following definitions are used in this paper:

- Item: piece of goods to be stored in a storage segment;
- Bin: set of storage segments having the same coordinate (height and length) in the retrieval face;
- Layer: set of storage segments located at the same depth. Each bin crosses the same number of layers;
- Row: number of bins in the height dimension;
- Column: number of bins in the length dimension.

3.2 In-deep class storage principle and objective validation

The main idea is to work on the content of the layers. A previous work on the classical class storage (Meghelli *et al.*, 2010) showed that retrieving an item in the second layer of a very well situated bin costs more time than retrieving an item in the first layer of a poorly situated bin. As a matter of fact, the first objective is to store the items in order to have the most popular items in the front layer. If this population of items is called class A, then this defines a new class-based storage method, called in-

deep class storage in this paper. This section intends to show the relationship between the tidiness of the rack and the performance of the retrieval of items in order to validate the objective of this paper.

Due to the complexity of the definition of the algorithms, this paper only presents the case of two classes. If this analysis is conclusive, a future work will evaluate the pertinence of increasing the number of considered classes.

3.2.1 Storage Indicator

To be able to quantify the tidiness of the racks, an indicator called Storage Indicator (SI) is defined. This indicator represents the number of times a class B item is stored before a class A item. Considering this definition, a perfectly tidied-up rack has $SI=0$. On the other hand, Table 1 gives the maximum SI that can be encountered for several classical rack configurations. This maximum is calculated for 1 bin, and then multiplied by the number of bins of the rack. Hence, for example, a rack of 10 rows, 10 columns and 10 layers full of items might have a $SI=500$ in the worst case. This is calculated considering there are 100 bins, each of them having 10 items stored in this exact order: Class B-A-B-A-B-A-B-A-B-A considering the first item in the front layer. Each bin has thus $RI=5$, and the whole rack $SI=500$.

<Please insert Table 1 here>

3.2.2 Objective validation experiments

The experiments deal with the evaluation by simulation of the average retrieval delay (ARD), considering the hypotheses described in section 4.1. Two separate configurations were created, one with a perfectly tidied-up rack (configuration A), and the other with totally randomly stored items (configuration B), both in the same rack type.

Each simulation runs the same way. First, a random Pareto distribution ($k=0.15=\text{constant}$) is generated to simulate the demand, different for each reference. According to this distribution, the rack is filled up with items. In configuration A, they are stored according to the number of items of each reference to be stored (the most represented first). As a matter of fact, class A is stored first, and thus SI cannot be else than equal to zero. In configuration B, items are stored in a random order, in the next randomly chosen bins with space available to store.

Then, a list of items to retrieve is determined according to the Pareto distribution. For each item, the one which will be retrieved the quickest is automatically chosen (according to next neighbor heuristic). The retrieval delay is recorded, and an average retrieval delay is calculated. 30 replications of each configuration were run in order to determine a confidence interval. Table 2 shows the mean of every ARD and SI recorded for each replication, and the half-width of the distribution (this value may be interpreted by saying "in 95% of repeated trials, the sample mean would be reported as within the interval sample mean \pm half width").

<Please insert Table 2 here>

As expected, the results show a larger retrieval delay for the randomly stored racks, which confirm the hypothesis of dominance of configuration A versus configuration B. The difference between the configurations is growing with the size of the rack, which can be explained by the growing number of references available at the same time in the rack, but is not so huge in percentage (5 to 15% in Table 2). This can be explained by the high probability to have a popular product in a low layer even though the storage is made randomly.

A second observation deals with SI for configuration B. In average, SI is about one third of its maximum theoretical value (Table 1) with a low half-width. This

shows that random storage almost never gives the worst possible storage of the whole rack.

This short experiment shows the relationship between the tidiness of the racks and the performance of the retrieval of items, measured through the average retrieval delay. Next section presents the algorithms which were created to implement these principles.

3.3 *In-deep class storage algorithms*

Two separate algorithms were developed, one dedicated to storage, the other one dedicated to retrieval. Both are triggered by:

- For the storage algorithm, detection of an item at the entrance station or restore station;
- For the retrieval algorithm, reception of a delivery request.

As a matter of fact, storage algorithm has to start with a priority rule (Figure 3), giving priority to class A product, then to the entrance station. A discrimination method between bins with the same potential was also defined. This method is based on the geographic localization of the bin in the rack, considering the restore and the entrance stations for storage algorithm, and the delivery station for the retrieval algorithm.

<Please insert Figure 3 here>

3.3.1 *Storage algorithm*

The objective of this algorithm is to make consecutive sequel of items with the same reference of class A (with the objective of dedicated bins), or sequel of class B items so that the SI of the rack would not be increased.

Thus, the algorithm (Figure 4) starts by seeking bins which have the corresponding reference located in the last layer. If none are found, then separate algorithms are built, according to the class of the item (Figures 5 and 6).

<Please insert Figure 4 here>

<Please insert Figure 5 here>

<Please insert Figure 6 here>

For class A items (Figure 5), empty bins are preferred, in order to create new dedicated bins. This preference is particularly useful for the initial storage of an empty rack. In steady state, empty bins are not so rare, because of the retrieval algorithm presented below. Then, the bins with the fewest class A items are promoted. This is meant to avoid to store a class A item in a too high number layer, and thus restore many class A items when retrieving this item. In case of several bins with the same properties, the bins with the lowest total of items (i.e. the lowest total of class B items) are promoted.

For class B items (Figure 6), the objective is to avoid storing them at an annoying place (i.e. in a bin where class A items could potentially be stored later by the algorithm). First, bins containing the maximum of the item's reference are promoted. It is very likely that this rule gives either a single solution or the whole rack (minus the full bins) as possibilities, as class B items are relatively poorly represented in the rack (rarely more than one or two items in the whole rack). In this group of possibilities, the second rule aims to forbid storing this item in a bin with a class A item in the last layer. Bins with the lowest SI and a class B item in the last layer are first chosen, then the bins with the fewest items and at least a class B, in order to avoid filling bins with only class A items.

3.3.2 *Retrieval algorithmx*

The objective of this algorithm is to retrieve an item with the satisfying reference in the lowest layer. There again, the algorithm is different according to the class of the desired item.

If it is a class A item which should be retrieved (Figure 7), the first action is to determine the lowest layer where the item is located. If several bins have this item in the first layer (numbered 0), the objective is to find those which also have an identical item in the second layer. Indeed, as it is a class A item, the aim is to ensure a good availability of the item in the low layers.

<Please insert Figure 7 here>

If it is not located in the first layer, then the bin which has the most class B products in the lower layers is chosen. Hence, these class B items will be restored at the back of the rack.

If it is a class B item (Figure 8), the first action is also to determine the lowest layer where the item is located. If several bins have this item in the first layer, the objective is to choose the item which will decrease the most the average layer of the class A items located in its bin.

<Please insert Figure 8 here>

On the other hand, as several items have to be restored at the back of the rack to be able to retrieve this item, the aim is to choose the bin with the least class A items before the desired item to be retrieved.

4 Simulation study

A simulation study was performed in order to evaluate the performance of the algorithms. To do so, a model was built on Rockwell Arena. Each algorithm is programmed in a stand-alone VBA function. The main objective of using Visual Basic was to be able to adapt these functions very easily in a real AS/RS command in

a later study. This section presents the assumptions made in the model, the validation process and the experiments that were run.

4.1 Model and assumptions

The model is built so that each parameter of the rack can be modified with a single variable: width, height, depth, load, Pareto distribution through its coefficient, speed of the machines, etc. It is also easily possible to switch between the use of the algorithms or a random rule, or to switch between different initial configurations of the items in the rack.

On the other hand, it was decided not to take into account several physical constraints. First, the load time, i.e. the time needed for the machines to load (or unload) an item from (to) a bin. Indeed, on real AS/RS, taking this time into account may or may not be crucial relatively to the speed of the machines. For example, in an industrial AS/RS with potentially fragile and heavy items, the speed of the machines is much less high than in a pharmaceutical AS/RS with generally very light items, with a potentially identical load time. Obviously, a high load time requires the items to be situated in a low layer, whatever bin the item is located in, whereas a low load time gives less importance to the layer and more to the geographical location of the bin. The algorithms that were presented try to optimize these two objectives, but the layer position first. As a matter of fact, in average, these algorithms are supposed to decrease the number of load times in the total retrieval delay. Thus, not taking into account the load time is the worst situation for the comparison of our algorithms, and this is why this hypothesis was considered.

Second constraint that is not taken into account is the availability of the storage machine. Indeed, this machine generally has a single capacity, which impacts the behavior of the restore queue in the restoring bin. Thus, between two separate

deliveries, there may be an important impact on the content of the rack because of the presence of several items in the restore queue. The objective is to measure the average retrieval delay, which requires being able to compare the values obtained for each delivery. As a matter of fact, an infinite speed was assigned for the storing machine, which is thus able to store all the items in no time.

The last assumption made in this paper deals with the distribution of item references in the rack. As stated before, the Pareto distribution was chosen, as being the classical probability distribution used to model the content of such stores. With this distribution, a reference $n^{\circ}x$ among a maximum number of references x_m has a cumulated probability of $1 - \left(\frac{x}{x_m}\right)^k$ to enter the store (k is the Pareto distribution coefficient). While filling up the rack, each time an item enters the entrance station, a random number (between 0 and 1) is fired and determines its reference. Table 3 shows an example of the effect of the use of this distribution on the content of a rack.

<Please insert Table 3 here>

What can be noticed is that:

- Some references are not present in the rack, as they are considered poorly popular. Of course, these references are not asked to be delivered;
- The references are indexed in a not strict decreasing order of popularity, i.e. the number of items of each reference is globally decreasing with the index, but two consecutive references might not follow this rule as it is randomly fired;
- The distribution of items references is stable along time, i.e. no new evaluation of the classification of the items is necessary.

Another assumption that is made here is that the rack always contains the same number of items, so that its load is constant. As a matter of fact, as soon as an item is delivered, it is instantly entering the entrance station, and thus instantly stored, due to the previous assumption. It is possible in the model to either fire a new reference for this item, or to leave the item with the same reference.

4.2 *Configuration of the experiments*

This section presents the experiments that were run to evaluate the performance of both algorithms presented before. As there are many parameters involved, having a more or less important influence on the performance of each algorithm, the aim of this section is to describe a classical configuration of the rack and of the behavior of the demand.

First, about the configuration of the rack, among those described in section 3.a, was chosen:

- Width: 25 bins;
- Height: 25 bins;
- Depth: 10 places.

This was chosen in order to avoid having too much side effects due to a low number of bins, a low number of references or a low number of items of each references. To be able to store and retrieve items from a maximum of bins without making a trivial problem, the average load of the system was set to:

- Average load: 75%.
- Then, about the Pareto distribution, several parameters have to be fixed:
- Limit between class A and class B: 20% of items;

- Pareto coefficient: $k=0.1$, corresponding to 20% of references representing 80% of the items ;
- Maximum number of references: $N_{ref}=3000$.

This distribution was chosen as one of the most classical, as stated in De Koster *et al.* (2007). Due to the objective of class storage, it is obvious that it is necessary to have a clear difference between class A and class B, and this Pareto coefficient is coherent with this objective.

Finally, it is necessary to define the initial state of the rack, as it is possible to start with a randomly or perfectly ($S_i=0$) tidied up rack. The choice here is to start with randomly stored items, so that it is possible to see at which point the developed algorithms enable a natural storage of the rack during its use.

4.3 *Experimental design*

Four separate experiment configurations were defined and are described on Table 4.

<Please insert Table 4 here>

For each of these configurations, a simulation study was run on Rockwell Arena. For each experiment configuration, 30 replications were made, each of them involving 100 000 retrievals, and the same 100 000 storages (same reference just after the item was retrieved). Between each replication, the initial random configuration of the rack was changed, as was the Pareto distribution. However, for a specific configuration of the rack and of the distribution, each experiment was executed so that the results could be more easily compared.

For each retrieval, the Retrieval delay (R_d) and the Storage indicator (S_i) were measured.

4.3.1 *Rd measurement*

The retrieval delay represents the time between the first move of the machine, always starting from its dwell point located at the middle of the rack face, and the time the item is delivered.

Obviously, there is no reason for two consecutive R_d measurements to be correlated (Figure 9 shows the time of the first 100 retrievals of a replication in configuration #1). This results in a very chaotic evolution of R_d along time. Thus, a study was lead in order to determine whether a linear regression or a simple average would be better to use to discriminate two storage/retrieval methods. Table 5 shows the results of this study.

<Please insert Figure 9 here>

<Please insert Table 5 here>

First point to notice, the correlation coefficients are almost equal to zero. This can be explained by the chaotic nature of the measurement. Then, the calculated slopes are also almost equal to zero, with a low standard deviation. This tends to show that the average of the average retrieval delay is enough to demonstrate the difference between the configurations. This is what will be used in the rest of this study.

4.3.2 *Si measurement*

On the other hand, when storage algorithm is turned on (experiments #3 and #4), the S_i evolution cannot be approximated with a linear regression, as it clearly has a decreasing evolution ending with a horizontal asymptote. As the measurements also involve a relatively high level of noise, the comparisons will be made on the response time (R_t).

In order to calculate R_t , a first step is to calculate the value of the horizontal asymptote (Figure 10). This is made with an average on the last 10 000 retrievals: the simulations were made on 100 000 retrievals to be sure these last 10 000 can be

considered as always being in the steady state. Then, response time at X% is defined as the earliest date when the storage indicator is below the level of the horizontal asymptote raised of X%.

<Please insert Figure 10 here>

5 Results and discussion

5.1 *Average retrieval delay*

Table 6 shows the average retrieval delay obtained from the simulations.

<Please insert Table 6 here>

The first conclusions that can be drawn from these results are:

- (1) The retrieval algorithm makes the average retrieval delay decrease of about 60 %.
- (2) In addition with the storage algorithm, the decrease is 5 % more important.
- (3) The standard deviation is bigger when the storage algorithm is turned on. This can be explained by the fact that, to try and optimize the retrieval delay of class A items, class B items are located in high number layers. As a matter of fact, class B products have a very high average retrieval delay, which explains the higher standard deviation.

5.2 *Storage indicator*

Table 7 shows the difference of storage indicator between the experiments

<Please insert Table 7 here>

The additional conclusions that can be drawn from these results are:

- (1) When the storage algorithm is turned off, the storage indicator does not decrease.

- (2) When the storage algorithm is turned on (experiments #3 and #4), the storage indicator tends to a very low level. This rack having 625 bins, $S_i=32$ means that at least 593 bins are perfectly tied up, and only 32 class B items are in ahead of class A items. It has to be noted that a level of $S_i=0$ is very often achieved on smaller racks.
- (3) The response time is a lot better in configuration #3, i.e. when the retrieval algorithm is turned off (around –85% compared to configuration #4). Once again, as this algorithm tries to optimize the retrieval delay, it tends to retrieve items from low layers. As a matter of fact, fewer items go back to the storing machine, and thus the storage is less efficient.

5.3 Calculations

This simulation study was lead on a total of 12 000 000 retrieval orders. On a common laptop (dual core at 2.3 GHz with 2 Gb of RAM), simulations lasted a total of 1250 minutes. This means that 160 retrieval/storage orders were treated each second by the Visual Basic algorithms. This is an important indicator for a future industrial implementation, as it seems very likely to be implementable on classical PLC controls.

6 Robustness of the algorithms

Many factors influence the performance of the algorithm: number and size of the bins, shape of the rack, number of different references, load of the rack and statistic distribution of the demand. A full study of these parameters' influence will be carried on in a next paper.

However, this section presents a robustness test aimed to validate the algorithms with a variable set of data. The parameter which was chosen is the shape

of the distribution of items in the rack and of the demand. As shown in Table 8, various Pareto distributions (differentiated with their Pareto coefficient) are compared with a uniform distribution.

<Please insert Table 8 here>

First column of Table 8 shows the probability that item X_1 is asked to be retrieved $P(X_1)$. This reference is the most frequent in the whole rack. When in the uniform case $P(X_1) = (N_{ref})^{-1}$, the probabilities for Pareto were calculated according to the definition of section 4.2. This indicator is expressed to ease the understanding of the evolution of parameter k .

With the same protocol as before, for each distribution, 30 replications of 100000 retrievals were simulated in order to mean the effects of the initial random storage and the random order of retrieval.

First, the results show that the algorithms are a lot more efficient when the most frequent items are very frequent (i.e. high k). This can be easily explained by the fact that, as the parameter k is rising, it is more and more possible to fill complete bins with items of the same reference. This tends to dedicated storage, which, when it is possible, is the close to be the best way of storage.

Finally, the results of the uniform distribution are far less interesting than Pareto. However, they are still better than those obtained in previous experiments with configurations #1 and #3, i.e. with a random retrieval (Table 6). This is interesting as it shows that, even if the distribution is not corresponding exactly to Pareto, the algorithms are still efficient.

7 Conclusion

This article defines a new storage/retrieval method in Flow-Rack AS/RS, called In-Deep Class Storage. The idea is to store the most popular items in the front layers of

each bin, so as to decrease the average retrieval delay of items in the rack. Two algorithms were developed to prove the feasibility of the implementation of this method.

A simulation study on a specific configuration of the rack was run, taking into account the stochastic behavior of the demand. This study showed a reduction of more than 60% of the average retrieval delay, compared to a random retrieval algorithm.

Two perspectives of this work will be developed shortly. First, a sensitivity analysis will be run, in order to determine the influence of each of the many variables on the performance of the algorithms. This analysis should explicit the optimal configurations of application of the algorithms.

Then, a third algorithm, dedicated to decrease the Storage Indicator during the machines free time, could be developed. This could be particularly relevant for applications where there is no night shift.

8 Acknowledgement

This work was partially funded by the EGIDE organization, via the TASSILI 11 MDU831 program (reference 24323VD).

9 References

- Allen, S. L. 1992. A selection guide to AS/R systems. *Industrial Engineering*, 24 (3), p. 28-31.
- Ashayeri, J., R. M. Heuts, M. W. T. Valkenburg, H. C. Veraarts, and M. R. Wilhelm, 2002. A geometrical approach to computing expected cycle times for zone-based storage layouts in AS/RS. *International Journal of Production Research*, 40(17), p. 4467-4483.
- Bozer Y. A. and J. A. White, 1984. Travel time models for automated storage/retrieval systems, *IIE Transactions*, 16(4), p. 329-338.
- Bozer Y. A., E. C. Schorn, G. P. Sharp, 1987. Geometric approaches to solve the Chebyshev travelling salesman problem, *Technical Report*, 87-29.
- Caron, F., G. Marchet, and A. Perego, 1998. Routing policies and COI-based storage policies in picker-to-part systems. *International Journal of Production Research*, 36(3), p. 713-732.
- Caron, F., G. Marchet, and A. Perego, 2000. Optimal layout in low-level picker-to-part systems. *International Journal of Production Research*, 38(1), p. 101-117.

- De Koster, R., T. Le-Duc, and K. J. Roodbergen, "Design and control of warehouse order picking: A literature review", *European Journal of Operational Research* 182 (2007) 481–501
- De Koster, R., T. Le-Duc, and Y. Yugang, 2008. Optimal storage rack design for a 3-dimensional compact AS/RS. *International Journal of Production Research*, 46(6), p. 1495-1514.
- Eldemir F., R. J. Graves and C. J. Malmberg, 2004. New cycle time and space estimation models for automated storage and retrieval system conceptualization, *International Journal of Production Research*, 42 (22), pp. 4767-4783.
- Eynan, A., M. J. Rosenblatt, 1994. Establishing zones in single-command class-based rectangular AS/RS. *IIE Transactions*, 26(1), p. 38–46.
- Gaouar N., Z. Sari, et N. Ghouali, 2005, Développement d'une Heuristique de Stockage/Déstockage pour un Système AS/R à Convoyeur Gravitationnel. 6e *Congrès International de Génie Industriel, (GI'2005)*, 7-10 Juin 2005, Besançon Micropolis, France.
- Gaouar N., Z. Sari, et N. Ghouali, 2006, Modélisation et implémentation d'une Heuristique de Stockage/Déstockage pour un Système AS/R à Convoyeur Gravitationnel. 6ème *Conférence Francophone de Modélisation et Simulation (MOSIM'06)*, 3-5 avril 2006, Rabat, Maroc.
- Graves, S. C., W. H. Hausman, and L. B. Schwarz, 1977. Storage-retrieval interleaving in automatic warehousing systems. *Management Science*, 23 (9), p. 935–945.
- Guenov M. and Raeside R., 1992. Zone shapes in class based storage and multicommand order picking when storage/retrieval machines are used. *European Journal of Operational Research*, 58, p. 37-47.
- Ha J-W and Hwang H., 1994. Class-based storage assignment policy in carousel system. *Computers industries Engineering*, 26 (3), p. 489-499.
- Hausman, W. H., L. B. Schwarz, and S. C. Graves, 1976. Optimal storage assignment in automatic warehousing systems, *Management Science*, 22 (6), p. 629–638.
- Le-Duc, T. and R. De Koster, 2004. Travel distance estimation in a single-block ABC storage strategy warehouse, In: Fleischmann and Klose (eds.) *Distribution Logistics: advanced solutions to Practical Problems*, p. 185-202 (Berlin: Springer Verlag).
- Le-Duc, T. and R. De Koster, 2005. Travel distance estimation and storage zone optimisation in a 2-block ABC-storage strategy warehouse. *International Journal of Production research*, 43(17), p.3561-3581.
- Le-Duc, T. and R. De Koster, 2007. Travel time estimation and order batching in a 2-block warehouse. *Report, European Journal of Operational Research*, 176(1), p. 374-388.
- Lee, H. F., 1997. Performance analysis for automated storage and retrieval systems. *IIE Transactions*, 29 (1), p.15-28.
- Meghelli, N., Z. Sari, O. Cardin, and P. Castagna, Evaluation de la performance d'un stockage par classes dans le contexte du stockage automatisé à accumulation dynamique, In *Actes de la 8e Conférence Francophone de Modélisation et Simulation (MOSIM'10)*, Hammamet, Tunisie, 2010.
- Muppani V. R. and G. K. Adil, 2008. A branch and bound algorithm for class based storage location assignment. *European Journal of Operational Research*, 189 (2) p. 492-507

- Park, B. C., R. D. Foley, J. A. White, and E. H. Frazelle, 2003. Dual command travel times and miniload system throughput with turnover-based storage. *IIE Transactions*, 35, p. 343–355.
- Park, B. C., R. D. Foley, and E. H. Frazelle, 2006. Performance of miniload systems with two-class storage. *European Journal of Operational Research*, 170, p. 144–155.
- Peters B. A., J. S. Smith and T. S. Hale, 1996. Closed form models for determining the optimal dwell point location in automated storage and retrieval systems, *International Journal of Production Research*, 34(6), p. 1757-1771.
- Roodbergen, K.J., 2001. *Layout and routing methods for warehouses*. Ph.D. Series, Erasmus Research Institute of Management (ERIM).
- Roodbergen K. J. and I. F. A. Vis, 2009. A survey of literature on automated storage and retrieval systems. *European Journal of Operational Research*, 194, p. 343-362
- Rosenblatt, M.J., and A. Eynan, 1989. Deriving the optimal boundaries for class-based automatic storage/retrieval systems. *Management Science*, 35(12), p. 1519–1524.
- Sari Z., C.Saygin, and N.Ghouali, 2005. Travel-Time Models for Flow-Rack Automated Storage and Retrieval Systems, *International Journal of Advanced Manufacturing Technology*, 25, p. 979-987.
- Yu Y. and R. De Koster, 2009. Designing an optimal turnover-based storage rack for a 3D compact automated storage and retrieval system, *International Journal of Production Research*, 47(6), p. 1551 - 1571.
- Yu Y. and R. De Koster, 2009b. Optimal zone boundaries for two-class-based compact three-dimensional automated storage and retrieval systems, *IIE Transactions*, 41(3), p. 194 – 208.
- Yu Y. and R. De Koster, 2011. Sequencing Heuristics for Storing and Retrieving Unit Loads in 3D Compact Automated Warehousing Systems, *IIE Transactions*, doi: 10.1080/0740817X.2011.575441.

Rows	Columns	Layers	Total Places	Max(SI)/bins	Max(SI)
5	5	5	125	2	50
10	10	10	1000	5	500
25	25	10	6250	5	3125

Table 1: Storage Indicator calculation examples for 100% full racks

Rows	Columns	Layers	Configuration	SI	SI Half-width	ARD mean	ARD Half-width
5	5	5	A	0	0	11.347	0.26678
5	5	5	B	16.033	1.2575	11.906	0.33434
10	10	10	A	0	0	31.896	0.26491
10	10	10	B	145.06	2.9266	35.725	0.60016
25	25	10	A	0	0	58.656	0.18138
25	25	10	B	900.10	7.0537	67.796	0.70187

Table 2: Objective validation experiments results

Reference	Quantity	Reference	Quantity	Reference	Quantity	Reference	Quantity	Reference	Quantity
1	64	21	7	41	1	61	1	81	1
2	50	22	4	42	2	62	1	82	2
3	27	23	8	43	3	63	0	83	1
4	12	24	4	44	3	64	2	84	1
5	14	25	1	45	0	65	2	85	0
6	11	26	3	46	4	66	0	86	1
7	5	27	2	47	0	67	0	87	0
8	6	28	1	48	5	68	2	88	0
9	4	29	1	49	0	69	2	89	1
10	10	30	3	50	2	70	1	90	0
11	10	31	3	51	1	71	0	91	0
12	1	32	1	52	2	72	1	92	0
13	4	33	2	53	2	73	1	93	0
14	9	34	2	54	1	74	1	94	1
15	5	35	3	55	0	75	2	95	0
16	11	36	1	56	0	76	2	96	1
17	4	37	0	57	0	77	0	97	0
18	1	38	2	58	1	78	0	98	1
19	7	39	2	59	1	79	2	99	0
20	6	40	2	60	0	80	3	100	1

Table 3. Pareto distribution of 100 different references in a 10x10x5 rack with 75% load

Experiment configuration	Storage method	Retrieval method
#1	Random	Random
#2	Random	Algorithm
#3	Algorithm	Random
#4	Algorithm	Algorithm

Table 4. Experiment configurations

Experiment	Slope of linear regression	Standard deviation of the slope	Correlation coefficient
#1	8.0×10^{-7}	5.2×10^{-5}	7.5×10^{-6}
#2	-5.2×10^{-6}	4.5×10^{-5}	1.2×10^{-5}

#3	1.4×10^{-5}	5.0×10^{-5}	7.5×10^{-6}
#4	-3.4×10^{-5}	4.7×10^{-5}	3.6×10^{-5}

Table 5. Linear regressions of retrieval delays

Experiment	Average Retrieval Delay	Standard deviation
#1	768.4	1.4
#2	300.4	2.2
#3	762.8	5.5
#4	259.5	5.1

Table 6. Average retrieval delays

Experiment	Average Si initial value	Horizontal asymptote	Rt(5%)		Rt(20%)	
			Mean	Standard deviation	Mean	Standard deviation
#1	650	651	#N/A	#N/A	#N/A	#N/A
#2	650	667	#N/A	#N/A	#N/A	#N/A
#3	633	31.75	6239	1623	4547	892
#4	643	43.39	41059	8431	28092	6750

Table 7. Storage indicator

Statistic distribution		$P(X_1)$	Retrieval delay	
			Average	Standard deviation
Uniform		0.03%	542.4	16.1
Pareto	k=0.01	8.98%	293.0	4.2
	k=0.05	10.33%	275.9	3.8
	k=0.1	12.15%	257.0	2.9
	k=0.2	16.21%	227.5	2.9
	k=0.5	29.83%	190.5	2.2
	k=1	50.02%	171.8	4.2

Table 8. Item statistic distribution study results

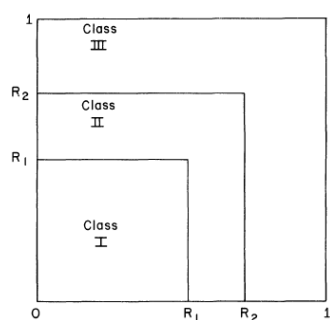


Figure 1. Example of (Hausman *et al.*, 1976) rack 3-class layout

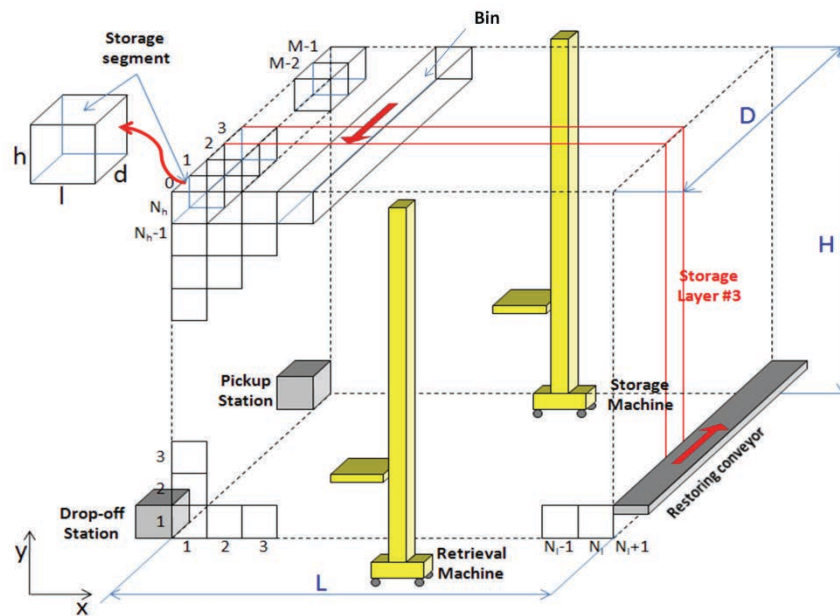


Figure 2. Configuration of the studied flow rack AS/RS

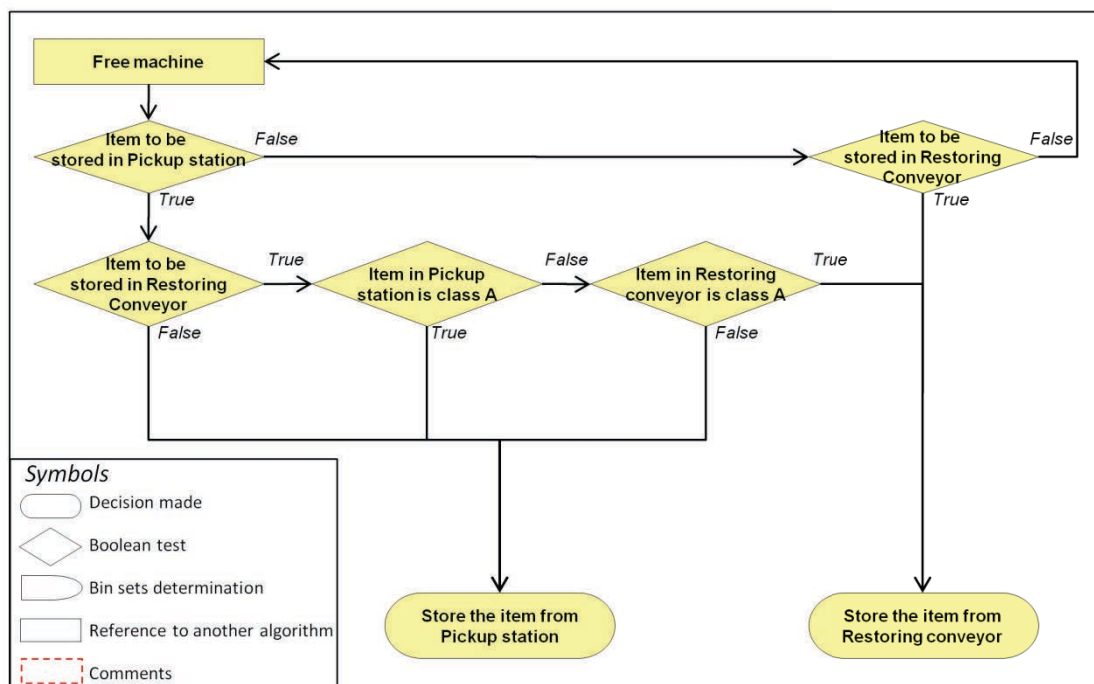


Figure 3. Priority rule

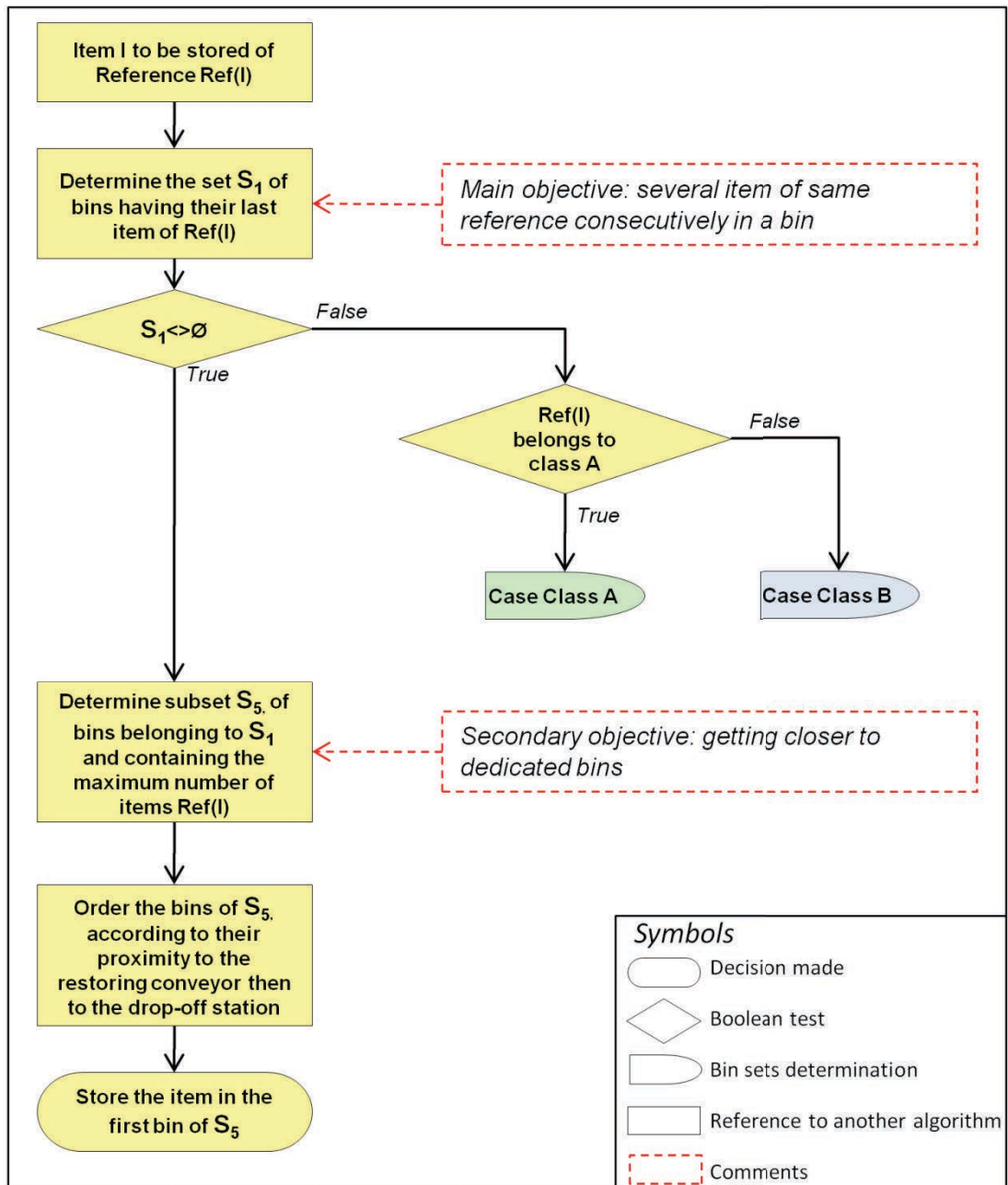


Figure 4. Storage algorithm main framework

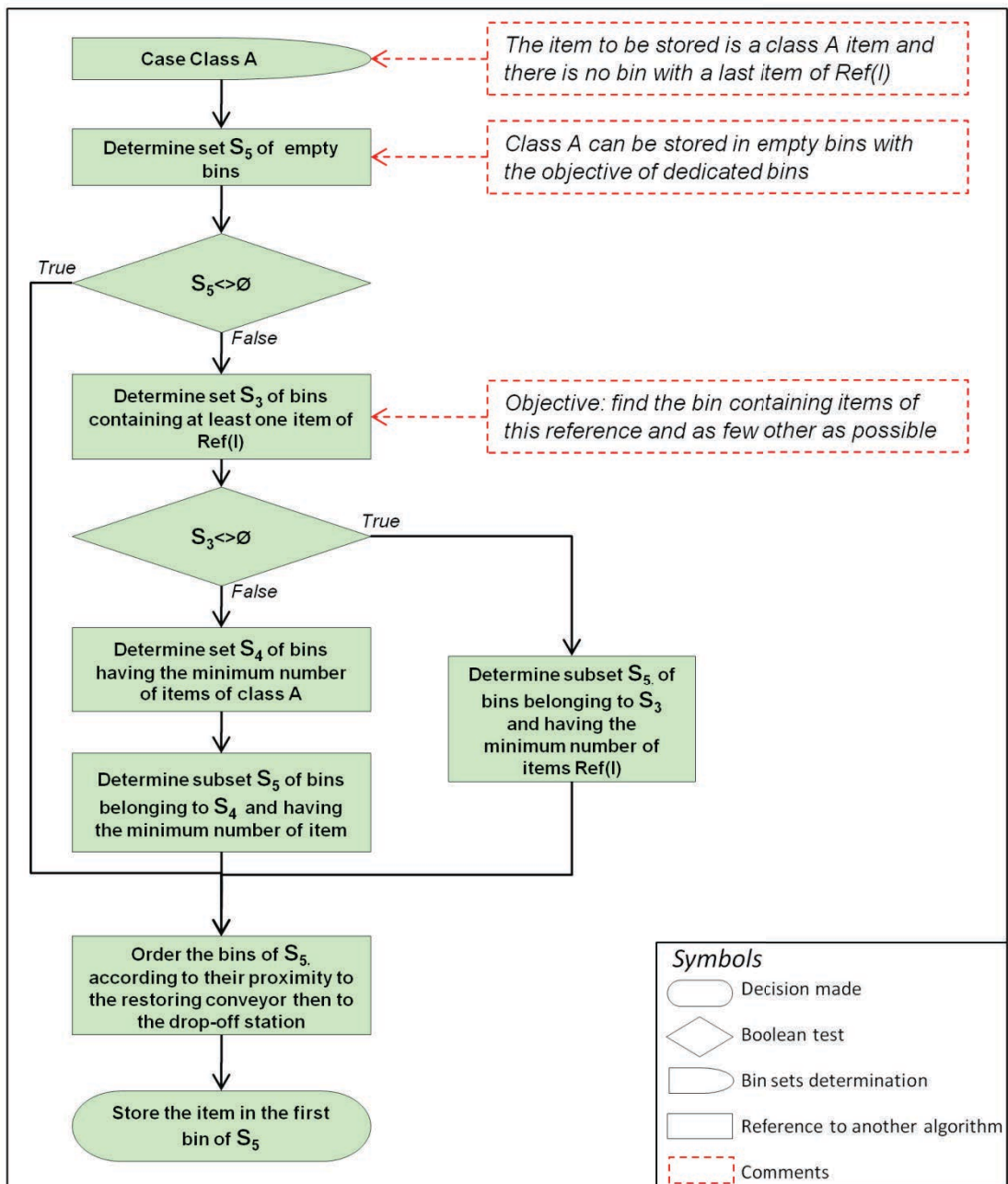


Figure 5. Storage algorithm for class A items

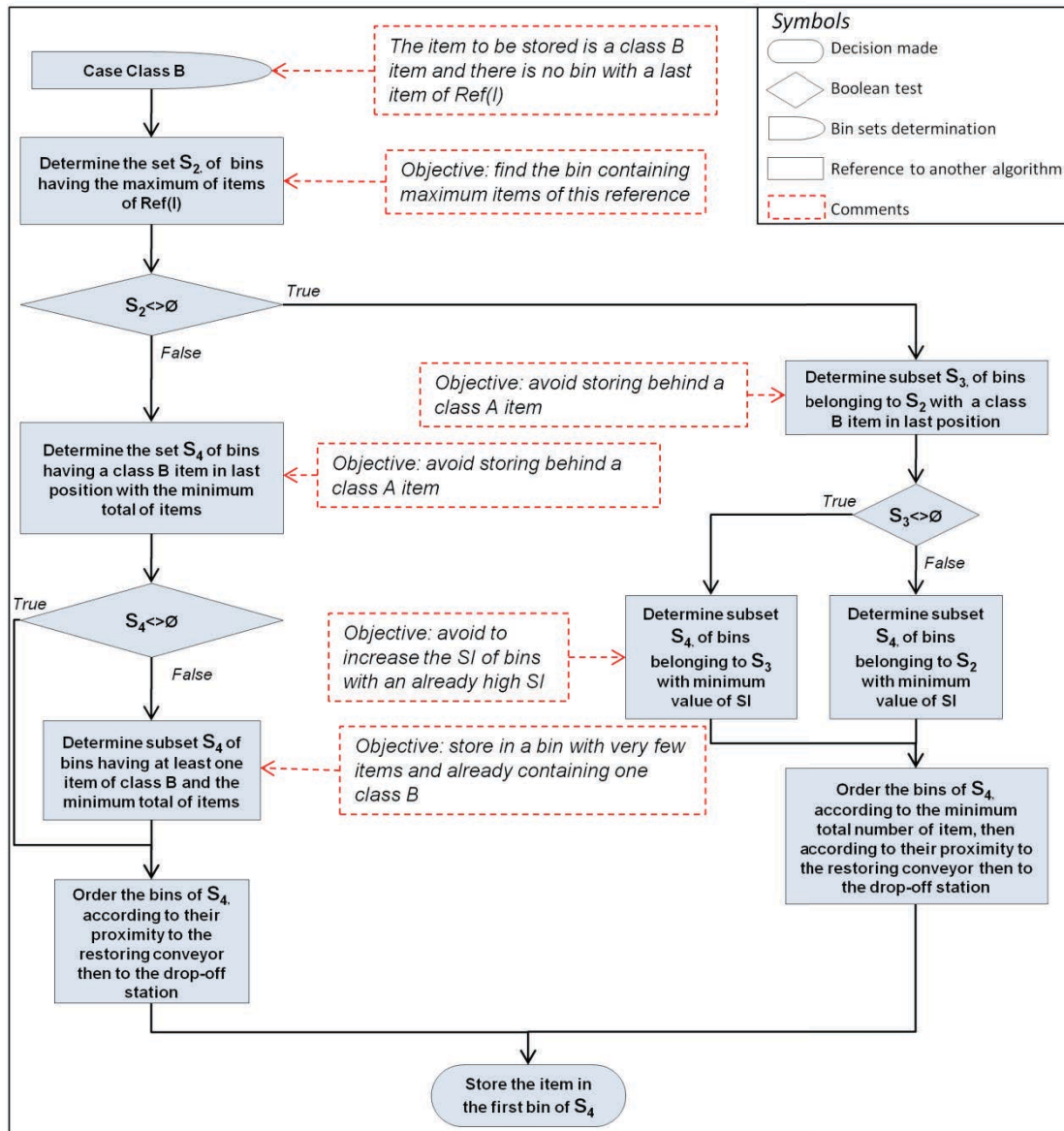


Figure 6. Storage algorithm for class B items

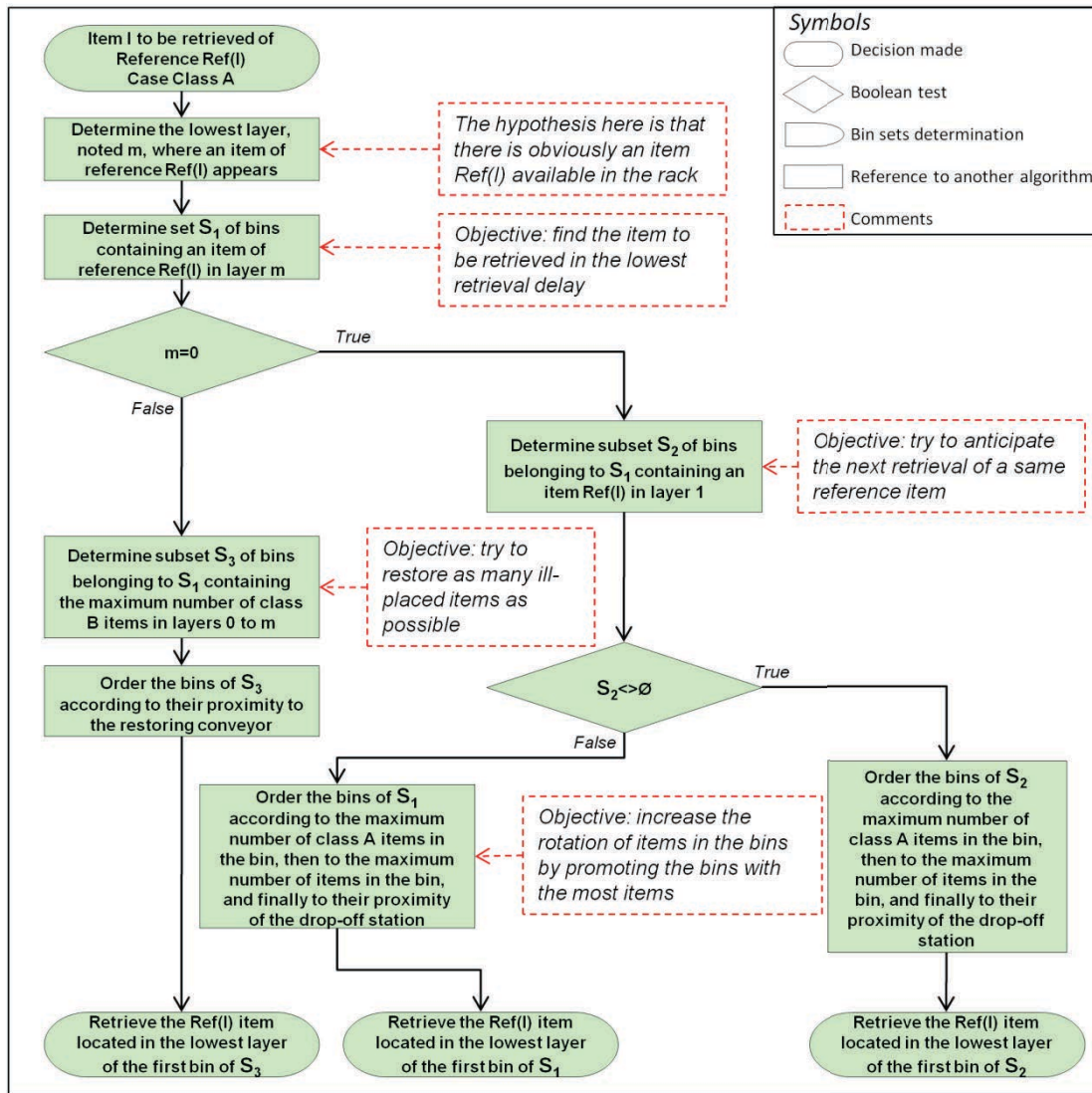


Figure 7. Retrieval algorithm for class A items

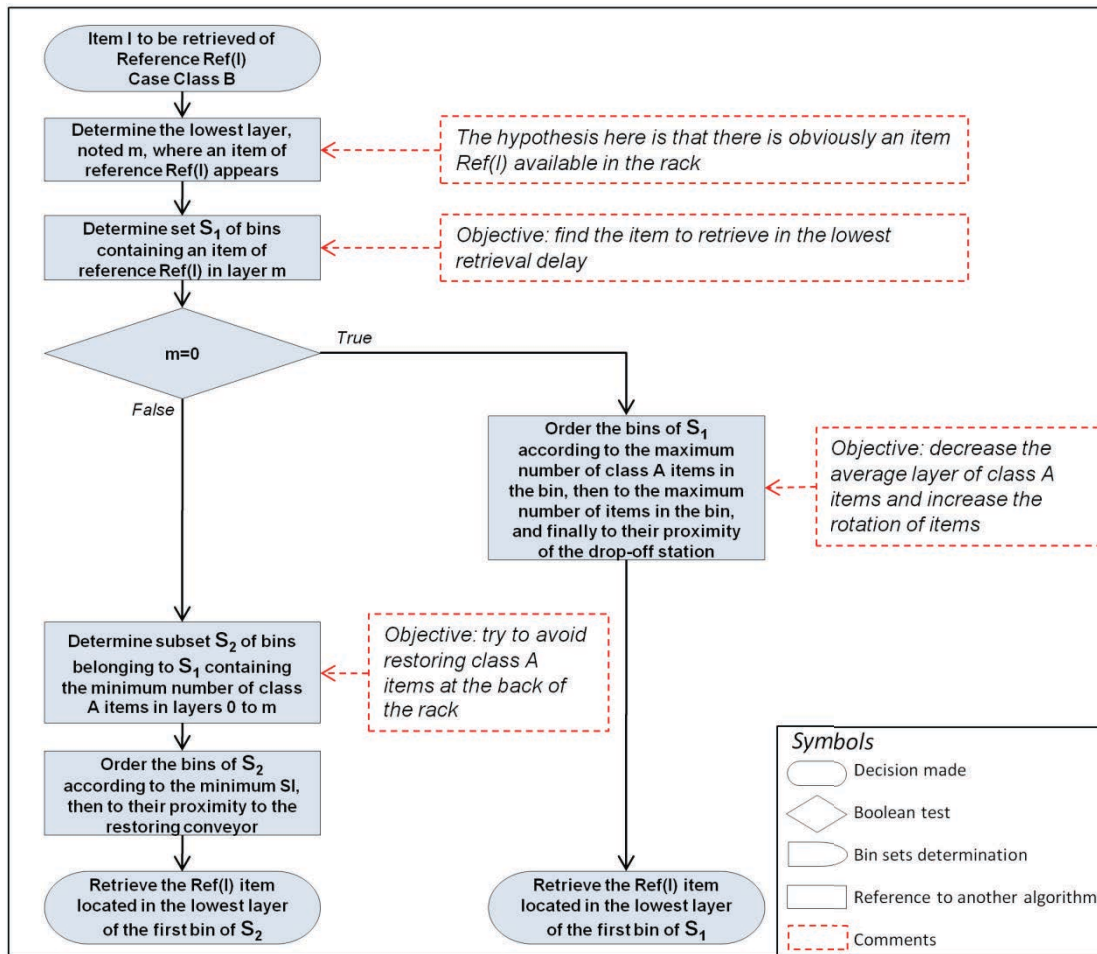


Figure 8. Retrieval algorithm for class B items

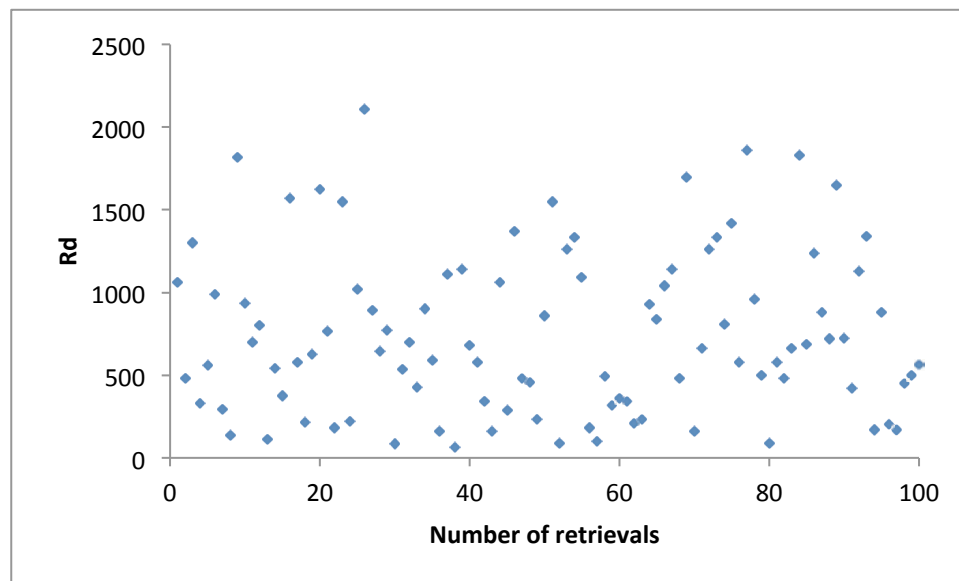


Figure 9. 100 first retrievals of a random replication in configuration (#1)

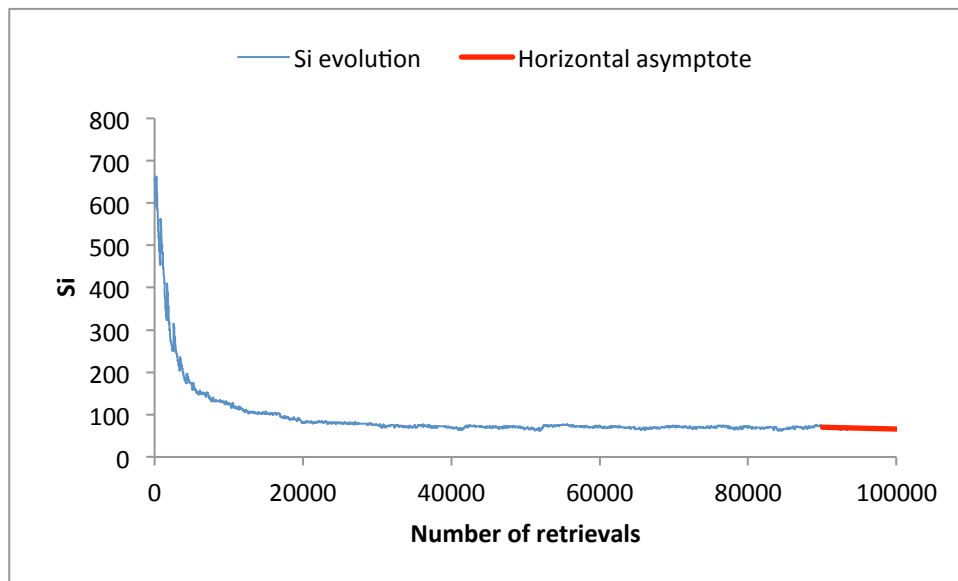


Figure 10. Definition of R_t on a random replication of configuration #4